Amendments to the Claims

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently amended) A method of processing encryption data in a computing entity, ~~said method characterized by~~ comprising:

assigning a memory means of said computing entity into a plurality of memory areas including a user space and a kernel space;

receiving encrypted data;

storing said encrypted data in ~~a first memory area of~~ said kernel space ~~computing entity~~, said ~~first memory area~~ kernel space being assigned for use by a kernel code of an operating system of said ~~first~~ computing entity;

writing said encrypted data stored in said ~~first memory area~~ kernel space into said user space, said user space being assigned for storing user application programs running on said computing entity ~~a second memory area associated with said computing entity~~;

decrypting said encrypted data stored in said user space ~~second memory area~~; and

writing said decrypted data from said ~~second memory area~~ user space to said kernel space ~~first memory area~~.

2. (Currently amended) A method as claimed in claim 1, wherein said ~~first memory area~~ kernel space is logically distinct from said ~~second memory area~~ user space.

3. (Currently amended) A method as claimed in claim 1, wherein said ~~first memory area~~ kernel space is configured to contain code of said operating system.

4. (Currently amended) The method as claimed in claim 1, wherein said ~~second memory area~~ user space is not used for storage of code of a kernel of said operating system.

5. (Currently amended) The method as claimed in claim 1, wherein said ~~step of~~ decrypting said encrypted data stored in said ~~second area~~ user space is carried out by an internet security protocol program resident in said ~~second memory area~~ user space.

6. (Currently amended) A method of processing encryption data in a computing entity ~~said method characterized by~~ comprising:

assigning a memory means of said computing entity into a plurality of memory areas including a user space and a kernel space;

storing decrypted data in ~~a first~~ said kernel space ~~memory area of said plurality of memory areas~~, said ~~first memory area~~ kernel space being assigned for use by a kernel code of an operating system of said ~~first~~ computing entity;

writing said stored data into ~~a second memory area of~~ said user space, said user space being assigned for storing user application programs running on said computing entity ~~plurality of memory areas associated with said first communicating entity~~;

encrypting said data stored in said ~~second memory area~~ user space; and

writing said encrypted data from said ~~second memory area~~ user space to said ~~first memory area~~ kernel space.

7. (Currently amended) A method as claimed in claim 6, wherein said ~~first memory area~~ kernel space is logically distinct from said ~~second memory area~~ user space.

8. (Currently amended) A method as claimed in claim 6, wherein said ~~first memory area~~ kernel space is configured to contain code of said operating system.

9. (Original) A method as claimed in claim 6, wherein said ~~second memory area~~ user space is not used for storage of code of a kernel of said operating system.

10. (Currently amended) A method as claimed in claim 8, further comprising ~~the step of~~ redirecting said encrypted data from said operating system in said ~~first memory area~~ kernel space to an encryption/decryption stack resident in said ~~second memory area~~ user space.

11. (Currently amended) A method as claimed in claim 9, comprising ~~the step of~~ directing said data to said ~~second memory area~~ user space from said ~~first memory area~~ kernel space.

12. (Currently amended) A method as claimed in claim 6, wherein said ~~step of~~ encrypting said data stored in said ~~second memory area~~ user space is carried out by an internet protocol security program stored in said ~~second memory area~~ user space.

13. (Currently amended) A method of processing encrypted data in a computing entity, said computing entity comprising[[;]] a processor[[;]] and a memory means[[,]] ~~wherein said memory means~~

~~is~~ divided into a user space and a kernel space ~~first and second memory areas~~, wherein said ~~first memory area~~ kernel space contains code of an operating system of said computer entity, said method comprising ~~the steps of~~:

receiving an encrypted data packet;

processing said data packet according to at least one packetization protocol of said operating system in said ~~first memory area~~ kernel space;

outputting said data packet to said ~~second memory area~~ user space;

processing said data packet according to a decryption algorithm in said ~~second memory area~~ user space; and

returning said processed data packet to said operating system in said ~~first memory area~~ kernel space.

14. (Currently amended) A digital computer configurable for transmitting digital data across a communications network, ~~said digital computer comprising;~~:

at least one microprocessor unit;

memory means, wherein said memory means is logically sub divided into at least a user space and a kernel space and characterized by further comprising[[;]] redirection means for writing data from said ~~first memory area~~ kernel space to said ~~second memory area~~ user space and from said ~~second memory area~~ user space to said ~~first memory area~~ kernel space; and

encryption means logically located within said ~~second memory area~~ user space, said encryption means being configurable for encrypting said digital data for transmission across the communications network.

15. (Currently amended) A digital computer configurable for receiving digital data transmitted across a communications network, ~~said digital computer~~ comprising:

at least one microprocessor unit;

a memory means, wherein said memory means is assigned into at least a user space and a kernel space ~~first memory area and a second memory area~~ and characterized by further comprising[[;]] a redirection means for writing data from said ~~first memory area~~ kernel space to said ~~second memory area~~ user space and from said ~~second memory area~~ user space to said ~~first memory area~~ kernel space; and

decryption means logically located within said ~~second memory area~~ user space for decrypting data stored in said ~~second memory area~~ user space, said decryption means being configurable for decrypting said data.

16. (Currently amended) A digital computer as claimed in claim 15, wherein said redirection means comprises[[;]]:

a redirection layer; and

a port for interfacing said redirection layer and said an encryption means.

17. (Currently amended) A digital computer as claimed in claim 15, wherein said encryption means logically located within said ~~second memory area~~ user space comprises:

an internet protocol security stack; and

a database configurable to contain key data for encrypting said data.

18. (Currently amended) A digital computer as claimed in claim 15, wherein said encryption means logically located within said ~~second memory area~~ user space comprises:

a plurality of internet protocol security stacks; and

a plurality of data bases configurable to contain key data for encrypting said data, wherein each data base of said plurality of databases contains a same key data.

19. (Original) A digital computer as claimed in claim 18, wherein said plurality of internet protocol security stacks are generated using a plurality of computing languages.

20. (Original) A computing entity comprising:

a data processing means;

a memory means;

an operating system having a set of kernel code, containing a communications protocol stack code;

a plurality of encryption and decryption means;

a first area of said memory means being assigned to said operating system code;

a second area of said memory means being assigned to said plurality of encryption and decryption means, said second memory area being sub-assigned into a plurality of compartmented memory areas within said second memory area, wherein individual ones of said encryption and decryption means are resident in corresponding respective ones of said plurality of compartmented memory areas.

21. (Original) The computing entity as claimed in claim 20, further comprising a director means resident in said first

memory area, said director means arranged to input data from and output data to said communications protocol stack.

22. (Original) The computing entity as claimed in claim 21, wherein said director means sends a plurality of data streams to said plurality of compartmented memory areas and receives a plurality of data streams from said plurality of compartmented memory areas.

23. (Original) The computing entity as claimed in claim 21, wherein said director means operates to send a corresponding respective data stream to each of said plurality of compartmented memory areas, and receive said corresponding respective data stream from said corresponding compartmented memory areas.

24. (Original) The computing entity as claimed in claim 20, further comprising a plurality of ports resident in said first memory area, there being at least one said port per said compartmented memory area, said port positioned between a said corresponding respective compartmented memory area, and a director means for directing data streams to and from said plurality of compartmented memory areas, said director means being resident in said first memory area.

25. (Original) The computing entity as claimed in claim 20, comprising a plurality of data processor means, said plurality of data processors providing processing capability for carrying out data processing operations within said plurality of compartmented memory areas.

26. (Currently amended) A method of encryption processing a plurality of packet data streams between first and second layers of a communications protocol stack, ~~said method~~ comprising:

receiving a first said data packet stream from a first layer of said protocol stack in a kernel memory space that is assigned for use by a kernel code of an operating system comprising said protocol stack ~~first memory area~~;

sending said data packet stream to a first compartmented memory area of a user memory space that is logically distinct from said kernel memory space;

running an encryption process on said first data packet stream in said first compartmented memory area for encryption or decryption of said data packet stream;

returning said processed data packet stream from said first compartmented memory area to a second layer of said communications protocol stack in said kernel memory space ~~first memory area~~;

receiving a second packet data stream from a said first or second layer of said communications protocol stack in said kernel memory space ~~first memory area~~;

sending said second data packet stream to a second compartmented memory area of said user memory space;

encryption processing said second data packet stream in said second compartmented memory area for encryption or decryption of said data packet stream; and

returning said processed second packet data stream to the other one of said first or second said layers of said communications protocol stack in said kernel memory space ~~first memory area~~,

wherein said first compartmented memory area is assigned to said first process[[,]] and said second compartmented memory

area is assigned to said second process~~, and said first memory area is assigned to an operating system of said computing entity~~.

27. (Currently amended) The method as claimed in claim 26, comprising ~~the step of~~ running a plurality of processes in a said compartmented memory area, for processing a single said packet data stream.